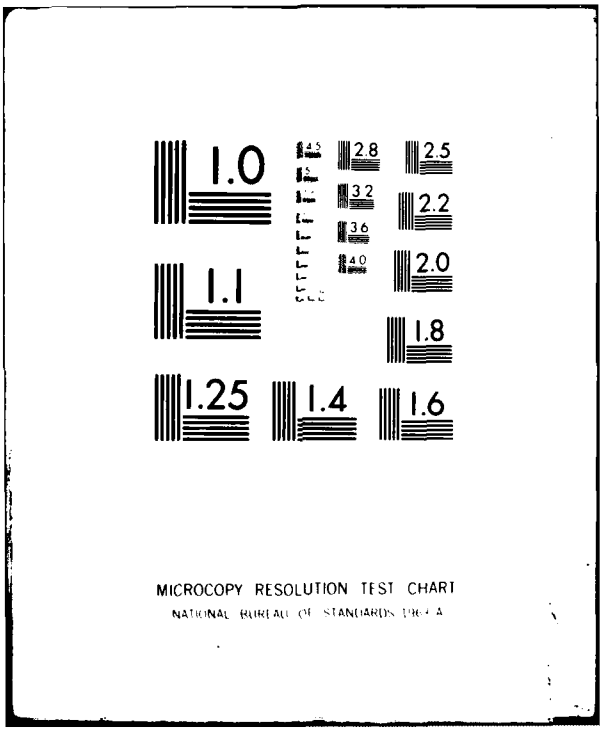
AD-A114 857 HARVARD UNIV CAMBRIDGE MA AIKEN COMPUTATION LAB F/G 12/2
A LOGARITHMIC TIME SORT FOR LINEAR SIZE NETWORKS.(U)
MAY 82 J H REIF, L G VALIANT N00014-80-C-0674
UNCLASSIFIED TR-13-82 NL

END
DATE FILMED
6 82
DTIC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO. 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): A Logarithmic Time Sort for Linear Size Networks	5. TYPE OF REPORT & PERIOD COVERED: Technical Report 6. PERFORMING ORG. REPORT NUMBER: TR-13-82
7. AUTHOR(s): John H. Reif, Leslie G. Valiant	8. CONTRACT OR GRANT NUMBER(s): N00014-80-C-0674
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Harvard University, Cambridge, MA 02138	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Office of Naval Research, 800 North Quincy Street, Arlington, VA 22217	12. REPORT DATE: May, 1982 13. NUMBER OF PAGES: 24
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office): same as above	15. SECURITY CLASS. (of this report) 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

unlimited

This document has been approved for public release and sale; its distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

unlimited

This document has been approved for public release and sale; its distribution is unlimited.

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

sorting networks, parallel sort, probabilistic algorithm, Cube Connected Cycles

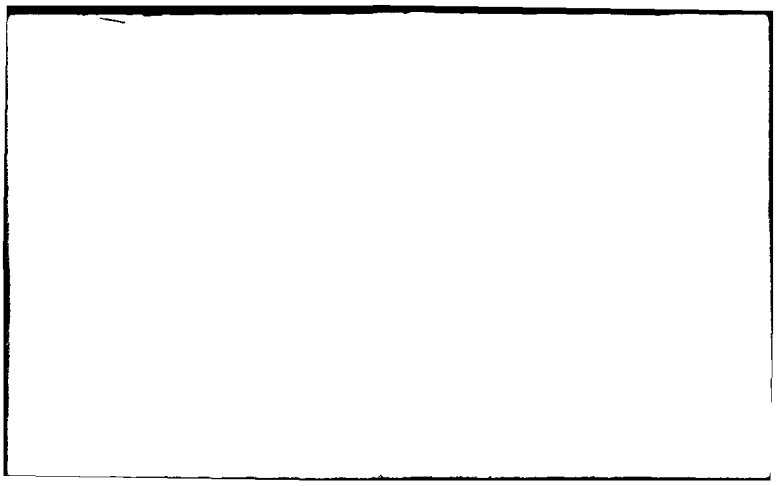
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

see reverse side

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. We give a probabilistic algorithm for sorting on constant valence networks of N nodes such as the cube-connected cycles. We prove that for any input set of O(N) keys, the algorithm's execution time is greater than a constant times log N with vanishingly low likelihood. Since this constant factor is small our parallel sorting algorithm may be practical to implement.



A LOGARITHMIC TIME SORT FOR LINEAR SIZE NETWORKS

John H. Reif
Leslie G. Valiant

TR-13-82

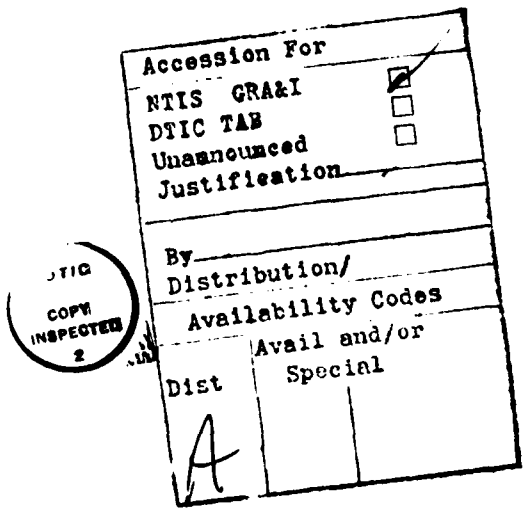
May, 1982

A Logarithmic Time Sort for Linear Size Networks

John H. Reif* and Leslie G. Valiant

Aiken Computation Laboratory
Division of Applied Sciences
Harvard University, Cambridge, Massachusetts

ABSTRACT. We give a probabilistic algorithm for sorting on constant valence networks of N nodes such as the cube-connected cycles. We prove that for any input set of O(N) keys, the algorithm's execution time is greater than a constant times log N with vanishingly low likelihood. Since this constant factor is small our parallel sorting algorithm may be practical to implement.



*This work was supported by the National Science Foundation Grant NAS-MCS79-21024 and the Office of Naval Research Contract N00014-80-C-0674.

1. INTRODUCTION

1.1 The Sorting Problem on a Network

This paper concerns the problem of parallel *sorting* on a fixed connection *network* $G = (V,E)$ of N nodes $V = \{v_0,\ldots,v_{N-1}\}$. Each node $v_i \in V$ is initially input a set X_i of c_0 distinct keys. Thus the set $X = X_0 \cup \ldots \cup X_{N-1}$ of all keys input is of size $c_0 N$. We assume a relation $<$ total ordering X. The network *sorts* X by routing each key $x \in X$ to node v_j where $j = \lfloor \mathrm{rank}(x)/c_0 \rfloor$ and $\mathrm{rank}(x) = |\{x' \in X | x' < x\}|$. Thus each sorting problem on network G can be viewed as a distributed *routing problem*. Each key $x \in X$ is considered a *packet* which must be routed from its *initial location* v_i, where $x \in X_i$, to the *destination node* v_j where the index $j = \lfloor \mathrm{rank}(x)/c_0 \rfloor$ must also be computed distributively.

1.2 Assumptions

We assume each node $v_i \in V$ contains a single sequential processor with local storage for $O(\log N)$ packets. These processors execute synchronously. At a single step each processor may make an elementary operation such as a key comparison, and cause transmission of a packet across each departing edge. Each processor uses randomization in the sense of Rabin [76] and Solovay and Strassen [77]. It is allowed on each step to choose random bits independently of the probabilistic choices of any other processor.

With these assumptions, the routing required to sort on network G requires time at least its *diameter* $\max_{u,v \in V}\{d | d$ is the length of the shortest path from u to $v\}$. If G has constant valence, then G has diameter $\Omega(\log N)$. Thus $\Omega(\log N)$ is a lower bound for the time to sort on any constant valence network of N processes.

1.3 FLASHSORT

Our FLASHSORT algorithm for parallel sorting uses a probabilistic divide and conquer technique similar to the popular sequential sorting algorithm QUICKSORT of Hoare [62]. QUICKSORT is popular because it can be practically implemented and is very fast. Sedgewick [75] shows that QUICKSORT takes expected time less than $cN \log N$ for a small constant c, given N randomly permuted input keys. FLASHSORT has similar advantages which we feel will lead to its practical utilization on distributed networks.

FLASHSHORT is executed in 4 phases sketched below:

I. (Random Routing) Route each key $x \in X$ to a randomly chosen node.

II. (Splitter Directed Routing) Choose a random key $\sigma \in X$. Use it to split $X - \{\sigma\}$ into disjoint subsets $\{x \in X | x < \sigma\}$ and $\{x \in X | \sigma < x\}$. Route these two disjoint subsets to disjoint subnetworks, and recursively apply II. This gives a rough sort of the keys X into disjoint subsets of small cardinality.

III. (Rank Computation) Compute $rank(x)$ for each key $x \in X$.

IV. (Rank Directed Routing) Route each $x \in X$ to node $v_{\lfloor rank(x)/c_0 \rfloor}$.

1.4 Organization and Results

Section 2 defines the CCC network and a related network CCC^+ on which we implement FLASHSORT. Section 3 describes the random routing in phase I and a probabilistic strategy for routing in phase IV proposed by Valiant and Brebner, [81]. Section 4 gives the details of the splitter-directed routing in phase II. Section 5 describes the rank computation in phase III. Sections 6-11 provide a probabilistic analysis of phases II

and III; we show that for any c, c_0 above a small number, $\exists c' \geq 1$ such that phases II and III take time less than $c \log N$ with probability greater than $1 - N^{-c'}$. Similar bounds on time have been proved by Aleliunas [82] and Upfal [82] for the packet routing in phases I and IV. Thus we conclude that on any given input, FLASHSORT achieves asymptotic optimal time $c \log N$, for a small constant c, on all but a vanishingly small number of executions. Section 12 describes some modifications to FLASHSORT which make it more practical when c_0 is small.

1.5 Previous Work

All previous algorithms for sorting N keys on a constant valence fixed connection network of N processors require time $\Omega(\log N)^2$. The parallel sorting algorithm of Batcher [68] achieves this time bound on various N node constant valence networks such as the CCC of Preparata and Vuillemin [81].

For less realistic models of computation faster algorithms are known. Several years ago J. Wiedermann observed that QUICKSORT takes time $c \log N$ with high likelihood on a parallel decision tree model with N processors. Reischuk [81] has a similar result for parallel random access machines.

Our algorithm follows the randomized routing ideas introduced in Valiant [82]. In the proofs heavy use is made of the critical path technique developed by Aleluinas [82] and Upfal [82].

2. NETWORK DEFINITIONS

Fix some number $n \geq 1$. This section defines two constant valence networks derived from the hypercube of dimension n. These networks have the same *node set* $V = \{(\alpha,i) \mid \alpha \in \{0,1\}^n,\ i \in \{0,\ldots,n-1\}\}$ with cardinality $N = n2^n$. For each $v \in V$ let $address(v) = \alpha$ and $stage(v) = i$ where $v = (\alpha,i)$. Let $\alpha[i]$ be the i-th bit of α and let $\alpha' = EXT(\alpha,i)$ be identical to α except $\alpha'[i]$ is the complement of $\alpha[i]$. Let edge $(u,v) \in V \times V$ be *internal* if address(v) = address(u) or *external* if address(v) = EXT(address(u),stage(u)+1). Also, let (u,v) be *forward* if stage(v) = (stage(u) + 1) mod n, *static* if stage(v) = stage(u), or *reverse* if stage(v) = (stage(u) - 1) mod n.

The Cube Connected Cycles (CCC_n) network of Preparata and Vuillemin [81] has node set V and exactly all forward-internal edges, reverse-internal edges, and static-external edges. For technical reasons this paper will assume a network previously defined in Upfal [82] which we call the CCC_n^+ network. It has node set V and exactly all forward-internal edges, reverse-internal edges, and forward-external edges. Thus the CCC_n and CCC_n^+ differ only with respect to the stage portions of external edges. Clearly any algorithm requiring routing on the CCC_n^+ network can be simulated on the CCC_n network with at most a factor of 2 time increase. (Since the transmission in CCC_n^+ of a packet x across a forward-external edge (u,v) can be simulated in CCC_n by transmission of x across forward-internal edge (u,w) followed by static-external edge (w,v).)

Note the CCC_n and CCC_n^+ are both naturally related to the hypercube H_n of dimension n. Intuitively, for each $\alpha \in \{0,1\}^n$ the set of nodes $\{u \in V \mid address(u) = \alpha\}$ can be considered to be a "supernode" of H_n. Each such "supernode" is connected by external edges to n other "supernodes" $\{v \in V \mid address(v) = EXT(\alpha,i)\}$ for $i = 0,\ldots,n-1$.

3. PACKET ROUTING ON THE CCC_n^+

This section briefly describes the probabilistic packet routing algorithm of Valiant and Brebner [81] as applied to the CCC_n^+ by Upfal [82]. This routing is required to implement phases I and IV of FLASHSORT.

We require that each node $v \in V$ contain for each departing edge e a queue Q_e of packets to be transmitted across edge e. Each node also contains its address and stage posted as local variables.

Let X be the set of c_0N packets to be routed, where each packet $x \in X$ is initially at a given node $I_x \in V$ and we wish x to be routed to given destination node $D_x \in V$. The algorithm has two phases:

A. (Random Routing) route x from I_x to a node $R_x \in V$ with random address.

B. (Fixed Destination Routing) route x from R_x to D_x.

The random routing of x in Phase A is accomplished by repeating for n stages the transmission of x across a randomly chosen departing forward edge (i.e., transmit x across the forward-internal edge or forward-external edge with equal probability). Phase B repeats for n stages the following: if x is currently at node $v \neq D_x$ with $j = \text{stage}(v) + 1$ and $\text{address}(v)[j] = \text{address}(D_x)[j]$, then x is transmitted across the forward-internal edge departing v and otherwise x is transmitted across the forward-external edge departing v. This takes the packets to the correct addresses. Finally, they are pipelined to the correct nodes in the cycles.

We have not yet specified the management of the queues of packets at each node. Suppose the *priority* of packet $x \in X$ is assigned to be the number of stages of phases A and B so far accomplished, and we allow packet x to be transmitted from each node $v \in V$ only after all packets of lower

priority are transmitted from v. Aleliunas [82] and Upfal [82] show:

THEOREM 3.1. *For any* c *above a small constant number,* $\exists c' > 1$ *such that the execution time of phase* A *and* B *exceeds* cn *with probability at most* $N^{-c'}$.

4. SPLITTER DIRECTED ROUTING

This section describes the splitter directed routing in phase II of FLASHSORT. Let $X[\lambda] = X$ be the set of c_0N keys input to FLASHSORT, where λ is the empty string. We index certain subsets of X by $\{0,1\}^{<n+1>}$, the set of binary strings of length at most n. Phase II is executed in stages $i = 0,\ldots,n-1$ where for each $\beta \in \{0,1\}^i$ if $X[\beta] \neq \emptyset$ we choose a random key $\sigma[\beta] \in X[\beta]$ which splits $X[\beta] - \{\sigma[\beta]\}$ into disjoint subsets $X[\beta 0] = \{x \in X[\beta] \,|\, x < \sigma[\beta]\}$ and $X[\beta 1] = \{x \in X[\beta] \,|\, \sigma[\beta] < x\}$. (The intention is to route the elements of $X[\beta]$ to the subcube specified by β.)

If $\sigma[\beta]$ is never defined for some $\beta \in \{0,1\}^{<n>}$ then phase II has a *blockage* and cannot be completed. This event is shown in Section 8 to have vanishingly low probability. If blockage does occur, then the execution of phase II will exceed a time limit c_4n (determined in Section 11 to hold with high likelihood assuming no blockage) and phases I and II must be re-executed. The probability of blockage in phase II the next time is independent of the first event of blockage in phase II. If there is no blockage, then phase II yields a rough sort of the keys X into a total of 2^{n+1} subsets, where 2^n disjoint subsets are of expected size less than c_0n with form $X[\alpha]$ where $\alpha \in \{0,1\}^n$ and there are also 2^n-1 singleton sets of form

$\{\sigma[\beta]\}$, where $\beta \in \{0,1\}^{<n>}$. Note that if $\alpha, \alpha' \in \{0,1\}^*$ where string α precedes α' in lexical order, then $x \in X[\alpha]$, $y \in X[\alpha']$ imply that $x < y$ in the total order.

We also define a recursive subdivision of the node set V, where we index subsets by binary strings in $\{0,1\}^{<n+1>}$. For each $\beta \in \{0,1\}^n$, let $V[\beta] = \{v \in V | \text{address}(v) = \beta \text{ and } \text{stage}(v) = 0\}$. For each $\beta \in \{0,1\}^{<n>}$ let $V[\beta] = \{v \in V | \beta \text{ is a prefix of address}(v) \text{ and } |\beta| = \text{stage}(v)\}$. Let the *root* $r[\beta]$ of $V[\beta]$ be the node with addres $\beta 0^{n-|\beta|}$ and stage $|\beta|$. Note that for $|\beta| < n-1$, $r[\beta]$ has a departing forward-internal edge entering $r[\beta 0]$ and also a departing forward-external edge entering $r[\beta 1]$. Also note that $V[\beta 0]$ and $V[\beta 1]$ are disjoint.

For each $i = 0,\ldots,n$ let $V_i = \{v \in V | \text{stage}(v) = i\}$.

We assume phase I routes each key $x \in X$ to a random node $v \in V_0$ and that the set of keys queued at each $v \in V_0$ are randomly ordered.

The *stage* of key $x \in X$ executing phase II is the stage of the node where x is currently visiting, except we define the stage of x to be n in the case x has just completed stage $n-1$ (and so just been routed from a node of stage $n-1$ to a node of stage 0). In the routing of phase II each of the keys $x \in X[\beta]$ visits only nodes in V whose address is prefixed by β, once x has passed stage $|\beta|$.

Initially each $v \in V[\beta]$ is said to be *inactive*, and transmits no keys of stage $|\beta|$. We *activate* $r[\beta]$ by choosing $\sigma[\beta]$ to be a particular key in $X[\beta]$ reaching the root node $r[\beta]$ so that $\sigma[\beta]$ is a random element of $X[\beta]$ (e.g., we may let $\sigma[\beta]$ be the first key of stage $|\beta|$ reaching $r[\beta]$). By a method described below, a copy of $\sigma[\beta]$ is routed to each node $v \in V[\beta]$. Node v becomes *activated* when this copy reaches it. It is able to transmit keys of stage $|\beta|$ only after it has been activated.

Let $x \in X[\beta] - \{\sigma[\beta]\}$ be a key at stage $|\beta|$ and visiting a node $v \in V[\beta]$. The key x remains at node v until v has been activated. When v has been activated, if $x < \sigma[\beta]$ then x is transmitted across the forward-internal edge departing v to a node in $V[\beta 0]$, and otherwise if $x > \sigma[\beta]$ then x is transmitted across the forward-external edge departing v to a node in $V[\beta 1]$.

Thus after all keys in X have completed stage $n-1$, we have for each $\beta \in \{0,1\}^n$ routed the keys $X[\beta]$ to the node $(\beta,0)$.

Now we give the details of how, for any $\beta \in \{0,1\}^{<n>}$, copies of $\sigma[\beta]$ are routed to each node in $V[\beta]$. A copy s of $\sigma[\beta]$ is called a *splitter*. For technical reasons (e.g., so that we do not confuse the delays due to key routing with those due to splitter routing) a splitter is considered a different type of packet from a key. (N.B. The type {key, splitter} of a packet can be specified by a boolean flag attached to the packet.) The *stage* of a splitter s copied from $\sigma[\beta]$ is fixed to be $|\beta|$ and does not vary during routing.

The splitter routing for β begins at root node $r[\beta]$ when $\sigma[\beta]$ is chosen. Splitters s, s' copied from $\sigma[\beta]$ are transmitted across the two forward edges (i.e., the forward-internal edge and forward-external edge) departing from $r[\beta]$.

Suppose node $v \in V$ receives a splitter s from an edge e entering v. If e is a forward edge and $\text{stage}(v) > 0$ then v transmits a copy of s across each of the two forward edges departing v. If e is a forward-external edge, then v transmits splitter s across the reverse-internal edge departing v in addition. If e is a reverse-internal edge and $\text{stage}(v) > \text{stage}(s)$ then s is transmitted across the reverse-internal edge departing v.

THEOREM 4.1. *Each* $v \in V[\beta]$ *is activated in* $2(n-|\beta|-\ell(v))$ *steps after* $r[\beta]$ *has been activated (where* $\ell(v) = \max\{\ell \mid 0^\ell$ *is a suffix of* address(v)$\}$*).*

Proof. Recall that $\text{address}(r[\beta]) = \beta 0^{n-|\beta|}$ and note that $\text{address}(v) = \beta\gamma 0^{\ell(v)}$ where $\gamma \in \{0,1\}^k$ and $k = n-|\beta|-\ell(v)$. Splitters copied from $\sigma[\beta]$ are routed from $v_0 = r[\beta]$ on forward edges $(v_0,v_1),\ldots,(v_{k-1},v_k)$ and then on reverse-internal edges $(v_k,v_{k+1}),\ldots,(v_{2k-1},v_{2k})$. For $i = 1,\ldots,k$ let (v_{i-1},v_i) be the forward-internal edge departing v_{i-1} if $\text{address}(v_{i-1})[|\beta|+i] = \text{address}(v_i)[|\beta|+i]$ and otherwise let (v_{i-1},v_i) be the forward-external edge departing v_{i-1}. Thus $\text{address}(v_{2k}) = \text{address}(v_k) = \text{address}(v)$ and $\text{stage}(v_{2k}) = \text{stage}(v_k) - k$ $= (\text{stage}(v) + k) - k = \text{stage}(v)$. Hence $v_{2k} = v$. □

5. THE RANK COMPUTATION

This section describes the rank computation done in phase III of FLASHSORT.

We show:

THEOREM 5.1. *The rank computation can be done in time*

$$4n + 3\left[\max_{\beta\in\{0,1\}^n}(|X[\beta]|)\right].$$

Proof. We begin by sorting the keys of $X[\beta]$ for each $\beta \in \{0,1\}^n$. This easily can be done in time $2|X[\beta]|$ by a parallel bubble sort using the cycle of n nodes $\{(\beta,0),\ldots,(\beta,n-1)\}$ connected by internal edges.

Next we compute at root $r[\beta]$ the value $|X[\beta]|$, for each $\beta \in \{0,1\}^{<n>}$. This is done in n stages $i = 0,\ldots,n-1$ using the identity $|X[\beta]| = |X[\beta 0]| + |X[\beta 1]| + 1$ for each $\beta \in \{0,1\}^i$. Note that the two forward edges departing $r[\beta]$ enter both $r[\beta 0]$ and $r[\beta 1]$, so the roots form a binary tree of depth n. The required sums on this tree can be done in time $2n$ using carry-adder logic [Tung, 72] to stream the bits of partial sums up the tree from nodes $r[\beta 0]$ and $r[\beta 1]$ to node $r[\beta]$ for each $\beta \in \{0,1\}^{<n>}$.

Finally, we compute rank by the following rule:

PROPOSITION 5.1. $\mathrm{rank}(\sigma[\lambda]) = |X[0]| + 1$. *For each* $\beta \in \{0,1\}^{<n-1>}$, $\mathrm{rank}(\sigma[\beta 1]) = \mathrm{rank}(\sigma[\beta]) + |X[\beta 10]| + 1$ *and* $\mathrm{rank}(\sigma[\beta 0]) = \mathrm{rank}(\sigma[\beta]) - |X[\beta 01]| - 1$. *For each* $\beta \in \{0,1\}^{n-1}$, *and each* $x \in X[\beta 1]$, $\mathrm{rank}(x) = \mathrm{rank}(\sigma[\beta]) + r + 1$ *where* $r = |\{x_1 \in X[\beta 1] \mid x_1 < x\}|$ *is the rank of* x *in* $X[\beta 1]$, *and for each* $x' \in X[\beta 0]$, $\mathrm{rank}(x') = \mathrm{rank}(\sigma[\beta]) + r' - |X[\beta 0]|$, *where* r' *is the rank of* x' *in* $X[\beta 0]$.

The additions required by this final computation can be done in time $2n + \max_{\beta \in \{0,1\}^n}(|X[\beta]|)$, again using carry-adder logic. □

6. CHERNOFF BOUNDS

This section gives some probabilistic inequalities, which will be useful in the sections following.

Let binomial variable b be the sum of N independent Bernoulli trials each with success probability p. Then the mean of b is $\bar{b} = Np$ and the probability that $b \geq m$ is $B(m,N,p) = \sum_{k=m}^{N} \binom{N}{k} p^k (1-p)^{N-k}$. The following inequality can be derived (see Angluin and Valiant [79]) from the bounds of Chernoff [52].

LEMMA 6.1. *For any* c, $0<c\leq 1$, $B((1+c)\bar{b},N,p) \leq \exp(-c^2\bar{b}/2)$, *and* $B((1-c)\bar{b}-1,N,p) \geq 1-\exp(-c^2\bar{b}/3)$.

Let g be the sum of N independent geometric random variables $g_1,\ldots,g_N$ with $\text{Prob}(g_i=k)=p(1-p)^k$ for $k\geq 0$. Then g has mean $\bar{g}=N(1-p)/p$. Let $G(m,N,p)$ be the probability $m\leq g$.

LEMMA 6.2. *For each* $c\geq 1$, *if* $m=c\bar{g}$ *then* $G(m,N,p) \geq 1-(1+(c-1)(1-p))^{m+N}/c^m$.

Proof. For any $z\in[1,1/(1-p))$, Chernoff [52] shows $G(m,N,p)$ is upper bounded by z^{-m} times the generating function of g, which is $(p/(1-(1-p)z))^N$. Our lemma follows from the case $z=c/(1+(c-1)(1-p))$. □

7. THE DISTRIBUTION OF KEYS ROUTED IN PHASES I AND II

For each $i=0,\ldots,n$ and $v\in V_i$, let $X(v,i)\subseteq X$ be the set of keys visiting node v at stage i of phase II. Fix a constant c_1 satisfying $0<c_1\leq 1$, and let $d_0=(1-c_1)c_0$.

Let $\mathcal{E}_0$ be the event $|X(v,0)|\geq d_0 n$ for all $v\in V_0$.

THEOREM 7.1. $\text{Prob}(\mathcal{E}_0)\geq 1-2^n f_0(n)$, *where* $f_0(n)=\exp(-c_1^2 c_0 n/3)$.

Proof. Fix some $v\in V_0$. Each key $x\in X$ has independent probability $|V_0|^{-1}=2^{-n}$ of being routed to node v by phase I. Thus the event $x\in X(v,0)$ is an independent Bernoulli trail with success probability 2^{-n}. $|X(v,0)|$ is therefore a binomial variable with parameters $|X|=c_0N$, 2^{-n} and mean $c_0 n$. Therefore $\text{Prob}(|X(v,0)|\geq d_0 n)\geq 1-B(d_0 n,c_0 N,2^{-n})\geq f_0(n)$ by Lemma 6.1. Hence $\text{Prob}(\mathcal{E}_0)\geq 1-|V_0|f_0(n)=1-2^n f_0(n)$, since $|V_0|=2^n$. □

Fix constant $d_1 = (1+c_1)c_0$. Let $\hat{\mathscr{E}}_0$ be the event $|X(v,i)| < d_1 n$ for all $i = 0,\ldots,n$ and $v \in V_i$.

THEOREM 7.2. $\text{Prob}(\hat{\mathscr{E}}_0) \geq 1 - N\hat{f}_0(n)$, *where* $\hat{f}_0(n) = \exp(-c_1^2 c_0 n/2)$.

Proof. Fix some $v \in V$, where $0 \leq i < n$. Since each key $x \in X$ is routed in phase I to a random node in V_0, its route in phase II is also random, so the event $x \in X(v,i)$ is upper bounded by an independent Bernoulli variable with success probability $|V_i|^{-1} = 2^{-n}$. $|X(v,i)|$ is upper bounded by a binomial variable with parameters $|X| = c_0 N$, 2^{-n} and mean $c_0 n$. Therefore $\text{Prob}(|X(v,i)| \geq d_1 n) \leq B(d_1 n, c_0 N, 2^{-n}) \leq f_1(n)$ by Lemma 6.1. Hence $\text{Prob}(\hat{\mathscr{E}}_0) \geq 1 - |V| f_1(n) = 1 - N f_1(n)$, since $|V| = N$. □

From Theorems 3.2 and 7.2 it follows

COROLLARY 7.2. *The rank computation of phase III takes time at most* $(4+3d_1)n$ *with probability at least* $1 - 2^n \hat{f}_0(n)$.

8. ACTIVATION PROBABILITIES

Let $\mathscr{E}_1$ be the event all nodes in V are eventually activated. Note that event $\mathscr{E}_1$ holds iff there is no blockage in phase II. Our calculation of the probability of $\mathscr{E}_1$ is complicated by the fact that the routing of a key x stops at root node $r[\beta]$ if $\sigma[\beta]$ is chosen to be x. For each $\beta \in \{0,1\}^{<n>}$ where $|\beta| > 0$ let $r^-[\beta]$ be the node of stage $|\beta| - 1$ such that there is a forward edge from $r^-[\beta]$ to $r[\beta]$ and the $|\beta|^{\text{th}}$ bit of $r^-[\beta]$ is 1. Since root node $r[\beta]$ has address $\beta 0^{n-|\beta|}$, we have:

PROPOSITION 8.1. *If* $x \in X(r^-[\beta], |\beta|-1)$ *then* x *visits no root node on any stage* $i < |\beta|$.

To simplify our calculation of the probability of $\mathscr{E}_1$ we make the following assumption about the procedure for choosing $\sigma[\beta]$. It ensures that the keys that are candidates for becoming splitters at a root have never been candidates at previous roots.

A1 For each $\beta \in \{0,1\}^{<n>}$ if $|\beta| = 0$ then $\sigma[\beta]$ is chosen to be a random key in $X(r[\lambda],0)$ and if $|\beta| > 0$ then $\sigma[\beta]$ is chosen to be the first key entering $r[\beta]$ from node $r^-[\beta]$.

THEOREM 8.1. $\text{Prob}(\mathscr{E}_1) \geq 1 - 2^n f_1(n)$, *where* $f_1(n) = (1-2^{-n-1})^{c_0 N}$.
$\leq \exp(-c_0 n/2)$.

Proof. For $i = 0,\ldots,n-1$ let $\mathscr{E}_1(\beta,i)$ be the event that for each $\beta \in \{0,1\}^i$, $X(r[\beta],i) \neq \emptyset$. For each $\beta \in \{0,1\}^i$ with $1 \leq i \leq n-1$ and each key $x \in X$, x visits $r^-[\beta]$ with probability 2^{-n}. Also, x reaches $r[\beta]$ by way of $r^-[\beta]$ with probability 2^{-n-1}. Hence, for any β, i the probability of $\mathscr{E}_1(\beta,i)$ exceeds

$$1 - (1-2^{-n-1})^{|x|} \geq 1 - f_1(n) .$$

This holds even for $\beta = \lambda$ and $i = 0$, when $(\lambda,0) \geq 1 - (1-2^{-n})^{|x|} \geq 1 - f_1(n)$. Hence the probability that all the events $\mathscr{E}_1(\beta,i)$ occur is greater than

$$1 - \sum_{i,\beta} f_1(n) \geq 1 - 2^n f_1(n).$$

By Theorem 4.1 if all these events occur then all nodes will be activated eventually. □

9. DELAY SEQUENCES

To simplify the calculation of the total time required in phase II we make the assumption:

A2 At the start of phase II the keys of $X(v,0)$ are assigned distinct *priorities* $\pi = 0,\ldots,|X(v,0)| - 1$ for each $v \in V_0$. Thereafter in phase II the priority $\pi(x)$ of each key $x \in X$ is fixed. Each active node $v \in V$ transmits keys of lowest priority first, so that no key x of priority $\pi(x)$ is transmitted from v before a key x' of priority $\pi(x') < \pi(x)$ is transmitted from v.

For each node $v \in V$ and integer $\pi \geq 0$, let the *key task* $[v,\pi]$ be the job of transmitting key packets of priority π from node v. For each $v \in V$ and i, $0 \leq i < n$, let the *splitter task* (v,i) be the job of transmitting a splitter of stage i from node v. (Recall that for a key $x \in X$ visiting node v, $stage(x) = stage(v)$, whereas for a splitter s visiting node v, $stage(s)$ is the stage of the node where the splitter was created.)

We define a precedence relation $\rightarrow$ between these tasks, where $\tau \rightarrow \tau'$ if the completion of some job in τ' must be delayed until some job in τ is completed. For each $u, v \in V$ and $\pi,\pi', i \geq 0$ we may have:

(1) $[v,\pi] \rightarrow [v,\pi+1]$ (since the transmission of a key x of priority $\pi+1$ may not be done before the transmission from v of keys of priority π).

(2) $[u,\pi] \rightarrow [v,\pi]$ for each forward edge (u,v) (since 1 step is required to transmit a key across edge (u,v) and the key's priority does not change).

(3) $[u,\pi] \rightarrow (v,i)$ where v is a root node $r[\beta]$, $u = r^{-}[\beta]$ for some $\beta \in \{0,1\}^i$ (since a key x of priority π may be chosen $\sigma[\beta] = x$).

(4) $(u,i) \rightarrow (v,i)$ where (u,v) is an edge of the CCC_n^+ network (since a splitter may be routed on any of the types of edges on the CCC_n^+).

(5) $(u,i) \rightarrow [v,\pi]$ where (u,v) is a reverse-internal edge and $stage(v) = i$. (Since a key of any priority π cannot be transmitted from v until node v has been activated by reception of a splitter of stage i from a node u).

(6) $[u,\pi'] \rightarrow [v,\pi]$ where v is a root node $r[\beta]$, $u = r^{-}[\beta]$ for some $\beta \in \{0,1\}^i$ (since a key of any priority π cannot be transmitted from v until a splitter has been created at v).

Let $\delta = \hat{\delta}_0\delta_0 \ldots \hat{\delta}_{n-1}\delta_{n-1}$ be a *delay sequence* if for $i = 0,\ldots,n-1$ δ_i is a (possibly empty) sequence of key tasks of stage i which are related by $\rightarrow$, and $\hat{\delta}_i$ is a (possibly empty) sequence of splitter tasks of stage i related by $\rightarrow$. Necessarily $\hat{\delta}_i$ is of the form:

$$(v_0,i),\ldots,(v_{k-1},i),(v_k,i),\ldots,(v_{2k-1},i)$$

where $i = stage(v_0)$, $i+1 = stage(v_{2k-1})$, (v_j,v_{j+1}) is a forward edge for $j = 0,\ldots,k-1$, (v_{k-1},v_k) is a forward-external edge, and $(v_k,v_{k+1}),\ldots,(v_{2k-2},v_{2k-1})$ are reverse-internal edges.

With each *execution* of phase II we can associate the set of delay sequences that describe the temporal sequences of causality in the obvious way. Thus two tasks are adjacent in such a sequence if the second one could not have been begun a time unit earlier than it was because the first task had not been completed yet. Our analysis will assume an "oracular"

version of the algorithm that only starts executing $[u,\pi]$ when all the keys of priority smaller than π that are to pass through u have already done so. The reader can verify by a little reflection that our analysis for the oracular algorithm upper bounds the performance of the actual algorithm. Another way is by augmenting the algorithm to enforce priorities as suggested by Upfal [82].

10. UPPER BOUNDS ON THE LENGTH AND NUMBER OF DELAY SEQUENCES

Let Δ be a set of all delay sequences occurring in an execution of FLASHSORT. Fix a constant c_2 satisfying $0 \leq c_2 \leq \min(1, d_0 - 4)$.

Let $\mathscr{E}_2$ be the event $|\delta| \leq (c_2+4)n$ for all $\delta \in \Delta$.

THEOREM 10.1. *Assuming* $\mathscr{E}_2$, $|\Delta| \leq 2^{d_2 n}$, *where* $d_2 = 1 + \lceil (c_2+4)\log 6 \rceil$.

Proof. Note that each $\delta \in \Delta$ can be completely specified by the start node in V_0, and a binary sequence of length $\lceil |\delta| \log 6 \rceil$, (since there are 6 types of pairs of tasks related by $\rightarrow$). By assumption $\mathscr{E}_2$, $|\Delta| \leq |V_0| 2^{\lceil (c_2+4)\log 6 \rceil n} = 2^{d_2 n}$. □

The Lemmas 10.1 and 10.2 proved in this section imply:

THEOREM 10.2. $\text{Prob}(\mathscr{E}_2 | \mathscr{E}_0 \wedge \mathscr{E}_1) \geq 1 - 2^{d_2 n} f_2(n)$, *where*

$$f_2(n) = (1 + (c_2 - 1)\exp(-\tfrac{1}{2}))^{(c_2+1)n} / c_2^{c_2 n}.$$

LEMMA 10.1. *With certainty,* $\Sigma_{i=0}^{n-1} |\hat{\delta}_i| \leq 2n$ *for any delay sequence* $\delta = \hat{\delta}_0 \delta_0 \cdots \hat{\delta}_{n-1} \delta_{n-1}$.

Proof. Recall from Section 4 that $\ell(v)$ is the length of the longest suffix of address(v) in O^*. If (u,v) is a forward edge and stage(u) < n-1, then $\ell(v) \geq \ell(u)$. Thus $\ell(v)$ never increases from successive key tasks $[u,\pi] \rightarrow [v,\pi']$ appearing in any δ_i. Let $\hat{\delta}_i = (v_0, i) \ldots (v_{2k-1}, i)$. By Theorem 4.1, $\ell(v_0) \geq \ell(v_{2k-1}) + |\hat{\delta}_i|/2$. This implies $\Sigma_{i=0}^{n-1} |\delta_i|/2 \leq n$. □

For any $\beta \in \{0,1\}^n$, let β_i be the prefix of β of length i. Let $\mathscr{P}$ be the event:

$$\sum_{i=0}^{n-1} \pi(\sigma[\beta_i]) \leq (c_2+1)n, \quad \text{for all} \quad \beta \in \{0,1\}^n .$$

LEMMA 10.2.

$$\text{Prob}(\mathcal{P} | \mathcal{E}_0 \wedge \mathcal{E}_1) \geq 1 - 2^n f_2(n) \quad .$$

Proof. Consider some fixed $\beta \in \{0,1\}^n$. Recall by assumption A1 that for each $i = 1,\ldots,n-1$ the root $r[\beta_i]$ chooses $\sigma[\beta]$ to be the first key entering $r[\beta_i]$ from node $r^-[\beta_i]$, and the $i+1$ bit of address$(r^-[\beta_i])$ is 1. Thus the set $U_i = \{(\gamma 10^{n-i}, 0) | \gamma \in \{0,1\}^{i-1}\}$ are those nodes of V_0 for which there is a path of $i-1$ forward edges to $r^-[\beta_i]$. Note that $U_i \cap U_j = \emptyset$ for $i \neq j$. By assumption $\mathcal{E}_0$, $|X(v,0)| \geq d_0 n$ for all $v \in V_0$. Let $X_{i,j} = \{x \in X(v,0) | v \in U_i$ and x has priority $j\}$.

Since each key in $X(v,0)$ has distinct priority, $|X_{i,j}| = 2^i$. Each $x \in X_{i,j}$ has independent probability $|U_i|^{-1}$ of getting routed to $r^-[\beta_i]$ and probability $1/2$ of transmission from $r^-[\beta_i]$ to $r[\beta_i]$. Hence for each $x \in X_{i,j}$ *the event* x *visits* $r[\beta_i]$ *is lower bounded by an independent* Bernoulli variable with success probability $|U_i|^{-1}/2$.

Let $\pi_i = \pi(\sigma[\beta_i])$. For each j, $0 \leq j \leq d_0 n$,
$\text{Prob}(\pi_i = j | \pi_i \geq j) = \text{Prob}(\exists x \in X_{i,j}$ where x visits $r[\beta_i]) \geq 1 - (1 - |U_i|^{-1}/2)^{|U_i|}$
$\geq 1 - \exp(-1/2)$. This implies $\text{Prob}(\pi_i = j) \geq p(1-p)^j$ where $p = 1 - \exp(-1/2)$. Hence for each $i = 0,\ldots,n-1$ the priority is upper bounded by an independent geometric variable with parameter p. We therefore can apply Lemma 6.2. □

LEMMA 10.3. *Assuming* $\mathcal{P}$, $\Sigma_{i=0}^{n-1} |\delta_i| \leq (c_2+2)n$ *for each* $\delta \in \Delta$ *where* $\delta = \hat{\delta}_0 \delta_0 \ldots \hat{\delta}_{n-1} \delta_{n-1}$.

Proof. For $i = 0,\ldots,n-1$, let π_i be the priority of the first task in δ_i. Then

$$\sum_{i=0}^{n-1} |\delta_i| \leq n + \sum_{i=0}^{n-1} \pi_i \leq n + (c_2+1)n$$

□

11. UPPER BOUNDS ON EXECUTION TIME OF PHASE II

Let T be the execution time of Phase II; we wish to derive on upper bounds for T, which hold with high likelihood. For each delay sequence $\delta \in \Delta$, let $T(\delta)$ be the time to execute the tasks of δ. Then

$$T = \max_{\delta \in \Delta} (T(\delta)) .$$

For key $x \in X$, let P_x be the sequence of nodes visited by x while stage$(x) < n$. For each delay sequence $\delta \in \Delta$, let $X_\delta = \{x \in X | v \in P_x$, $\pi = \pi(x)$ and $[v,\pi] \in \delta\}$; this is the set of keys whose transmission are tasks of δ.

Fix a constant c_3, satisfying $0 \leq c_3 \leq 1$, and let $d_3 = (c_3+1)(c_2+4)$. Let $\mathscr{E}_3$ be the event $|X_\delta| < d_3 n$ for all $\delta \in \Delta$.

LEMMA 11.1. $\text{Prob}(\mathscr{E}_3 | \mathscr{E}_2) \geq 1 - 2^{d_2 n} f_3(n)$ *where* $f_3(n) = \exp(-c_3^2(c_2+4)n)$.

Proof. Consider any key task $[v,\pi] \in \delta$ and key $x \in X$, with priority$(x) = \pi$. The event x visits v in phase II is upper-bounded by a Bernoulli variable with success probability 2^{-n}, independent of any other key. But there are at most 2^n keys in X of any given priority. Thus $|X_\delta|$ is upper-bounded by a binomial random variable with parameters 2^{-n} and $2^n|\delta|$. The latter is less than $2^n(c_3+4)$ by Assumption $\mathscr{E}_2$. By Lemma 6.1 therefore $\text{Prob}(|X_\delta| \geq d_3 n | \mathscr{E}_2) < f_3(n)$. Hence $\text{Prob}(\mathscr{E}_3 | \mathscr{E}_2) \geq 1 - |\Delta| f_3(n) \geq 1 - 2^{d_2 n} f_3(n)$ by Theorem 10.1. □

For each delay sequence $\delta \in \Delta$ and key $x \in X$ let $\tau(\delta,x) = \{[v,\pi] \in \delta | \pi = \pi(x)$ and $v \in P_x\}$. Since exactly one step is required

to process key x for each task $[v,\pi] \in \tau(\delta,x)$, we have $T(\delta) = \Sigma_{x \in X_\delta} |\tau(\delta,x)|$.

Fix a constant $c_4 \geq 1$.

THEOREM 11.2. $\text{Prob}(T \leq c_4 n | \mathcal{E}_2 \wedge \mathcal{E}_3) \geq 1 - 2^{d_3 n} \cdot f_4(n)$ *where* $f_4(n) = c_4^{-c_4 n} \cdot (1 + (c_4-1)/2)^{(c_4+d_3)n}$.

Proof. Let $\mathcal{E}_4$ be the event $|T(\delta)| \leq c_4 n$ for all $\delta \in \Delta$. Note that $\mathcal{E}_4$ implies $T \leq c_4 n$.

For each $x \in X_\delta$, we claim $|\tau(\delta,x)|$ is upper bounded by an independent geometric random variable g_x with parameter 1/2, so we can apply Lemma 6.2 to bound $\text{Prob}(|\tau(\delta,x)| \leq c_4 n | \mathcal{E}_3) \geq 1 - f_4(n)$, and by Theorem 10.1, $\text{Prob}(\mathcal{E}_4 | \mathcal{E}_2 \wedge \mathcal{E}_3) \geq 1 - |\Delta| f_4(n) \geq 1 - 2^{d_2 n} f_4(n)$.

Now we prove our claim. Fix some $x \in X$. Let $p_x = v_0,\ldots,v_m$ and $\pi = \pi(x)$. For each $i = 0,\ldots,m-1$, let (v_i,u_i) be the forward-internal edge departing v and let (v_i,w_i) be the forward-external edge departing v. Note that $u_i \in P_x$ iff $w_i \notin P_x$. The event $u_i \in P_x$ given $[v_i,\pi] \in \tau(\delta,x)$, has independent probability 1/2 for each $i = 0,\ldots,m-1$. Furthermore, if $v_i \in P_x$ and $[v_i,\pi] \in \tau(\delta,x)$ but $[v_{i+1},\pi] \notin \tau(\delta,x)$, then $[v_j,\pi] \notin \tau(\delta,x)$ for $j = i+1,\ldots,m$. Hence for $k \geq 0$, $\text{Prob}(|\tau(\delta,x)| = k) \leq 2^{-k-1}$ as claimed. □

Finally, we have:

THEOREM 11.3. *For* c_0 *above a small number,* $\exists c' \geq 1$ *such that* $\text{Prob}(T \leq c_4 n) \geq 1 - N^{-c'}$.

Proof.

$$\text{Prob}(T \leq c_4 n) \geq \text{Prob}(\mathcal{E}_0) \cdot \text{Prob}(\mathcal{E}_1) \cdot \text{Prob}(\mathcal{E}_2 | \mathcal{E}_0 \wedge \mathcal{E}_1) \cdot \text{Prob}(\mathcal{E}_3 | \mathcal{E}_2) \cdot \text{Prob}(T \leq c_4 n | \mathcal{E}_2 \wedge \mathcal{E}_3).$$

Theorems 7.1, 7.2, 10.2, 11.1, and 11.2 imply

$$\begin{aligned}\text{Prob}(T \leq c_4 n) &\geq (1-2^n f_0(n))(1-2^{n} f_1(n))(1-Nf_2(n))(1-2^{d_3 n} f_3(n))(1-2^{d_3 n} f_4(n)) \\ &\geq 1 - (2^n f_0(n) + 2^{n} f_1(n) + Nf_2(n) + 2^{d_3 n} f_3(n) + 2^{d_3 n} f_4(n)) \\ &\geq 1 - N^{-c'} \quad \text{for some } c' > 1. \end{aligned}$$

□

12. BLOCKADE AVOIDANCE

We prove in Theorem 8.1 that the probability of blockage is vanishingly low if c_0, the number of keys initially input at each node, is greater than a small constant number, say h. If $1 \leq c_0 < h$ then we can initially randomly route all keys to a subnetwork of the CCC_n with node set $V' = V[0^{\lceil \log h \rceil}]$ of cardinality $\leq N/h$. Then each node in V' will have on the average at least packets and we can execute FLASHSORT on this subnetwork.

A more practical method is to modify our FLASHSORT algorithm so that harmful blockage never occurs. Note that if $\sigma[\beta]$ is not defined because $X[\beta]$ is empty then the rank computation of Section 5 is still valid. The harmful case is when $\sigma[\beta]$ is not defined though $X[\beta]$ is nonempty. The

modified algorithm avoids this case altogether by making each member of $X[\beta]$ a viable candidate for becoming $\sigma[\beta]$. We define a new type of packet which we call a *candidate* and which, for routing purposes, is considered to be distinct from a key or a splitter. We route candidate packets for $\sigma[\beta]$ in essentially the opposite paths as for the splitters derived from $\sigma[\beta]$. Suppose a node $u \in V[\beta]$ receives a candidate packet t of stage $|\beta|$. If u is active or has previously received any key, splitter or candidate packet of stage $|\beta|$, then the current candidate packet is deleted. If u is not the root node $r[\beta]$ then the entering candidate packet t is transmitted across departing edge e (where if address(u) has suffix $0^{n-|\beta|}$, then e is the departing reverse-internal edge, and if address(u)[stage(u) + 1] = 0 then e is the departing forward-internal edge, and otherwise e is the departing forward-external edge). Finally, if u is the root node $r[\beta]$, then $\sigma[\beta]$ is chosen to be the entering candidate t, of stage $|\beta|$ and the splitter-creation process then proceeds as described in Section 4. It is easy to verify using a proof similar to Theorem 4.1 that if $X[\beta] \neq \emptyset$, then a candidate for stage reaches the root $r[\beta]$. Furthermore an argument similar to Lemma 10.1 shows that this candidate routing requires additional time on any delay path at most $2n$.

13. FURTHER WORK

A further paper, to appear, describes implementation of FLASHSORT on the shuffle-exchange network, multi-dimensional arrays, grids, and also hybrid networks of grids with CCC subnetworks. For these hybrid networks, FLASHSORT has the asymptotic optimal VLSI bit-complexity $AT^2 = O(N \log N)^2$ for sorting N keys (represented in binary) within time T (with high likelihood) and area $A \leq O(N^2)$.

REFERENCES

D. Angluin and L.G. Valiant. Fast probabilistic algorithms for Hamiltonian circuits and matchings. *J. of Comp. and Syst. Sci.* (1979), 155-193.

R. Aleliunas. Randomized parallel communication. Proc. of ACM Symp. on Principles of Distributed Computing, Ottawa, Canada (1982).

K. Batcher. Sorting networks and their applications. AFIPS Spring Joint Comp. Conf. 32 (1968), 307-314.

H. Chernoff. A measure of asymptotic efficiency for tests of hypothesis based on the sum of observations. *Ann. of Math. Stat.* 23 (1952), 493-507.

C.A.R. Hoare. QUICKSORT. *Computer J.* 5(1), (1962), 10-15.

F.P. Preparata and J. Vuillemin. The cube-connected cycles: A versatile network for parallel computation. *CACM* 24 (1981), 300-310.

M.O. Rabin. Probabilistic algorithms. In *Algorithms and Complexity*. J.F. Traub (ed.), Academic Press, New York, 1976.

R. Reischuk. A fast probabilistic parallel sorting algorithm. Proc. of 22nd IEEE Symp. on Foundations of Computer Science (1981), 212-219.

R. Sedgewick. Quicksort. STAN-C5-75-492, Dept. of Comp. Sci., Stanford Univ., May 1975.

R. Solovay and V. Strassen. A fast Monte-Carlo test for primality. *SIAM J. on Computing* 6 (1977), 84-85.

C. Tung. Arithmetic. In *Computer Science*. A.F. Cardenas, L. Presser, and M.A. Marin (eds.), Wiley-Interscience, New York, 1972.

E. Upfal. Efficient schemes for parallel communication. Proc. of ACM Symp. on Principles of Distributed Computing, Ottawa, Canada (1982).

L.G. Valiant. A scheme for fast parallel communication. *SIAM J. on Computing* 11:2 (1982), 350-361.

L.G. Valiant and G.J. Brebner. Universal schemes for parallel communication. Proceeding of the Thirteenth Annual ACM Symposium on Theory of Computing (1981), 263-277.


DATE
ILME
8